

Wazuh Server Health monitoring

ossec.conf

```
<!-- CPU, memory, disk metric -->
<localfile>
  <log_format>full_command</log_format>
  <command>echo $(top -bn1 | grep Cpu | awk '{print $2+$4+$6+$12+$14+$16}' | sed 's/,././g' ; free -m | awk 'NR==2{printf "%.2f\t\t\t\n", $3*100/$2 }' | sed 's/,././g' ; df -h | awk '$NF=="/" {print $5}' | sed 's/%//g' | sed 's/,././g')</com>
<alias>general_health_metrics</alias>
  <out_format>$(timestamp) $(hostname) general_health_check: $(log)</out_format>
  <frequency>30</frequency>
</localfile>

<!-- load average metrics -->
<localfile>
  <log_format>full_command</log_format>
  <command>uptime | grep load | awk '{print $(NF-2),$(NF-1),$(NF)}' | sed 's/,././g'</command>
  <alias>load_average_metrics</alias>
  <out_format>$(timestamp) $(hostname) load_average_check: $(log)</out_format>
  <frequency>30</frequency>
</localfile>

<!-- memory metrics -->
<localfile>
  <log_format>full_command</log_format>
  <command>free --bytes | awk 'NR==2{print $3,$7}'</command>
  <alias>memory_metrics</alias>
  <out_format>$(timestamp) $(hostname) memory_check: $(log)</out_format>
  <frequency>30</frequency>
</localfile>

<!-- disk metrics -->
<localfile>
  <log_format>full_command</log_format>
  <command>df -B1 | awk '$NF=="/" {print $3,$4}'</command>
  <alias>disk_metrics</alias>
  <out_format>$(timestamp) $(hostname) disk_check: $(log)</out_format>
  <frequency>30</frequency>
</localfile>
```

restart manager

local_decoder.xml

```
<!-- CPU, memory, disk metric -->
<decoder name="general_health_check">
  <program_name>general_health_check</program_name>
</decoder>

<decoder name="general_health_check1">
  <parent>general_health_check</parent>
  <prematch>ossec: output: 'general_health_metrics':\.</prematch>
  <regex offset="after_prematch">(\S+) (\S+) (\S+)</regex>
  <order>cpu_usage_%, memory_usage_%, disk_usage_%</order>
</decoder>
```

```

<!-- load average metric -->
<decoder name="load_average_check">
  <program_name>load_average_check</program_name>
</decoder>

<decoder name="load_average_check1">
  <parent>load_average_check</parent>
  <prematch>ossec: output: 'load_average_metrics':\.</prematch>
  <regex offset="after_prematch">(\S+), (\S+), (\S+)</regex>
  <order>1min_loadAverage, 5mins_loadAverage, 15mins_loadAverage</order>
</decoder>

<!-- Memory metric -->
<decoder name="memory_check">
  <program_name>memory_check</program_name>
</decoder>

<decoder name="memory_check1">
  <parent>memory_check</parent>
  <prematch>ossec: output: 'memory_metrics':\.</prematch>
  <regex offset="after_prematch">(\S+) (\S+)</regex>
  <order>memory_used_bytes, memory_available_bytes</order>
</decoder>

<!-- Disk metric -->
<decoder name="disk_check">
  <program_name>disk_check</program_name>
</decoder>

<decoder name="disk_check1">
  <parent>disk_check</parent>
  <prematch>ossec: output: 'disk_metrics':\.</prematch>
  <regex offset="after_prematch">(\S+) (\S+)</regex>
  <order>disk_used_bytes, disk_free_bytes</order>
</decoder>

```

restart manager

local_rules.xml

```

<group name="performance_metric,">

<!-- CPU, Memory and Disk usage -->
<rule id="100054" level="3">
  <decoded_as>general_health_check</decoded_as>
  <description>CPU | MEMORY | DISK usage metrics</description>
</rule>

<!-- High memory usage -->
<rule id="100055" level="12">
  <if_sid>100054</if_sid>
  <field name="memory_usage_%">^[8-9]\d|100</field>
  <description>Memory usage is high: $(memory_usage_%)%</description>
  <options>no_full_log</options>
</rule>

<!-- High CPU usage -->
<rule id="100056" level="12">
  <if_sid>100054</if_sid>
  <field name="cpu_usage_%">^[8-9]\d|100</field>
  <description>CPU usage is high: $(cpu_usage_%)%</description>
  <options>no_full_log</options>

```

```

</rule>

<!-- High disk usage -->
<rule id="100057" level="12">
  <if_sid>100054</if_sid>
  <field name="disk_usage_%">^[7-9]\d|100</field>
  <description>Disk space is running low: $(disk_usage_%)%</description>
  <options>no_full_log</options>
</rule>

<!-- Load average check -->
<rule id="100058" level="3">
  <decoded_as>load_average_check</decoded_as>
  <description>load average metrics</description>
</rule>

<!-- memory check -->
<rule id="100059" level="3">
  <decoded_as>memory_check</decoded_as>
  <description>Memory metrics</description>
</rule>

<!-- Disk check -->
<rule id="100060" level="3">
  <decoded_as>disk_check</decoded_as>
  <description>Disk metrics</description>
</rule>

</group>

```

- Rule ID 100054 is the base rule for detecting resource monitoring events.
- Rule ID 100055 is triggered when the memory utilized exceeds 80%.
- Rule ID 100056 is triggered when the CPU utilized exceeds 80%.
- Rule ID 100057 is triggered when the disk space used exceeds 70%.
- Rule ID 100058 is triggered when a CPU load average check is done.
- Rule ID 100059 is triggered when a memory metric check is done.
- Rule ID 100060 is triggered when a disk metrics check is done.

restart manager

Wazuh dashboard

Les champs personnalisés nouvellement ajoutés, `data.1minloadAverage`, `data.5min_loadAverage`, `data.15mins_loadAverage`, `data.cpu_usage%`, `data.memoryusage%`, `data.diskusage%`, `data.disk_free_bytes`, `data.disk_used_bytes`, `memory_used_bytes`, `data.memory_available_bytes` seront affichés comme des champs inconnus comme illustré dans la figure ci-dessous. Cela est dû au fait que le tableau de bord Wazuh peut ne pas reconnaître les nouveaux champs. Vous devez mettre à jour le modèle d'index (index pattern) dans Kibana pour inclure les nouveaux champs.

W.

Dashboards Ma...

Index patterns

wazuh-alerts-*

Recently viewed

server management

Rules

Decoders

CDB Lists

Status

Cluster

Statistics

Logs

Settings

Dev Tools

Ruleset Test

Security

Indexer management

Index Management

Snapshot Management

Sample Data

Dev Tools

Dashboard management

Dashboards Management

Reporting

Server APIs

App Settings

About

Undock navigation

Dashboards Management

Index patterns

Data sources

Saved objects

Advanced settings

wazuh-alerts-*

Time field: 'timestamp'

Default

This page lists every field in the **wazuh-alerts-*** index and the field's associated core type as recorded by OpenSearch. To change a field type, use the OpenSearch [Mapping API](#)

Fields (646)

Scripted fields (0)

Source filters (1)

Search

All field types

Name	Type	Format	Searchable	Aggregatable	Excluded
@timestamp	date				
@version	string				
GeoLocation.area_code	number				
GeoLocation.city_name	string				
GeoLocation.continent_code	string				
GeoLocation.coordinates	number				
GeoLocation.country_code2	string				
GeoLocation.country_code3	string				
GeoLocation.country_name	string				
GeoLocation.dma_code	number				

Rows per page: 10

<

1

2

3

4

5

...

65

>

Vérification

wazuh-alerts-*

Time field: 'timestamp'

Default

This page lists every field in the **wazuh-alerts-*** index and the field's associated core type as recorded by OpenSearch. To change a field type, use the OpenSearch [Mapping API](#)

Fields (4 / 672)

Scripted fields (0)

Source filters (0 / 1)

memor

All field types

Name	Type	Format	Searchable	Aggregatable	Excluded
data.memory_available_bytes	string				
data.memory_usage_%	string				
data.memory_used_bytes	string				
data.ms-graph.physicalMemoryInBytes	string				

Rows per page: 10

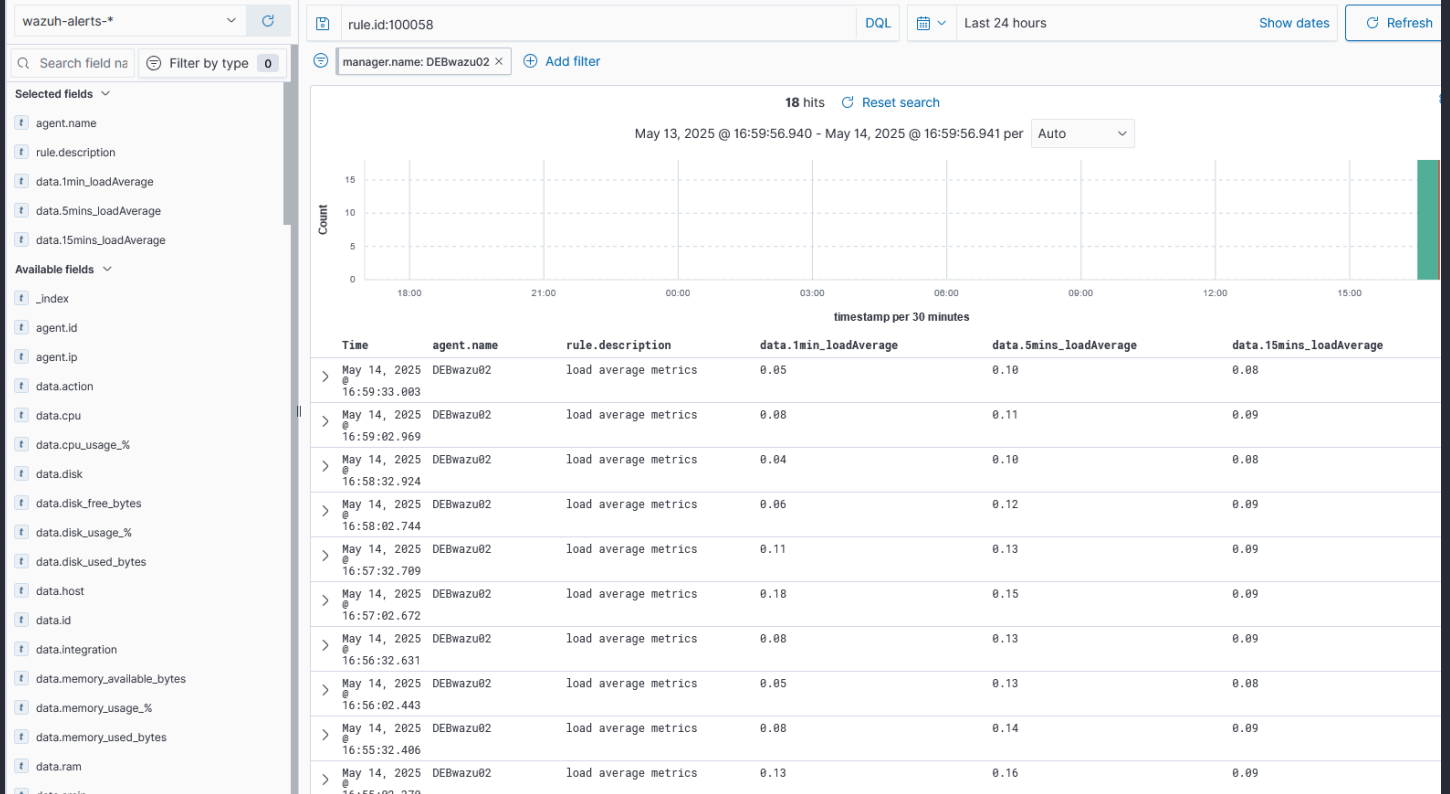
<

1

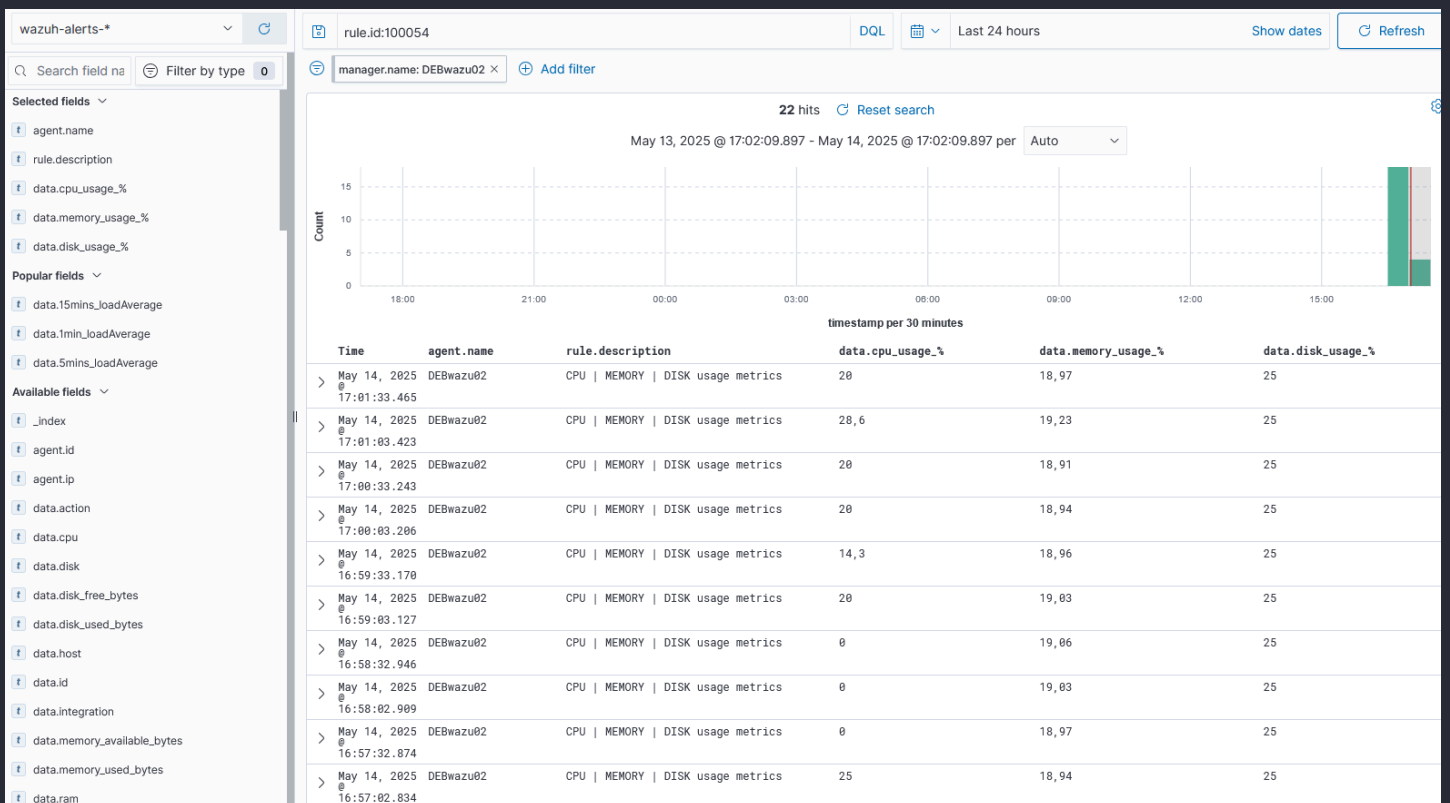
>

Alert visualisation

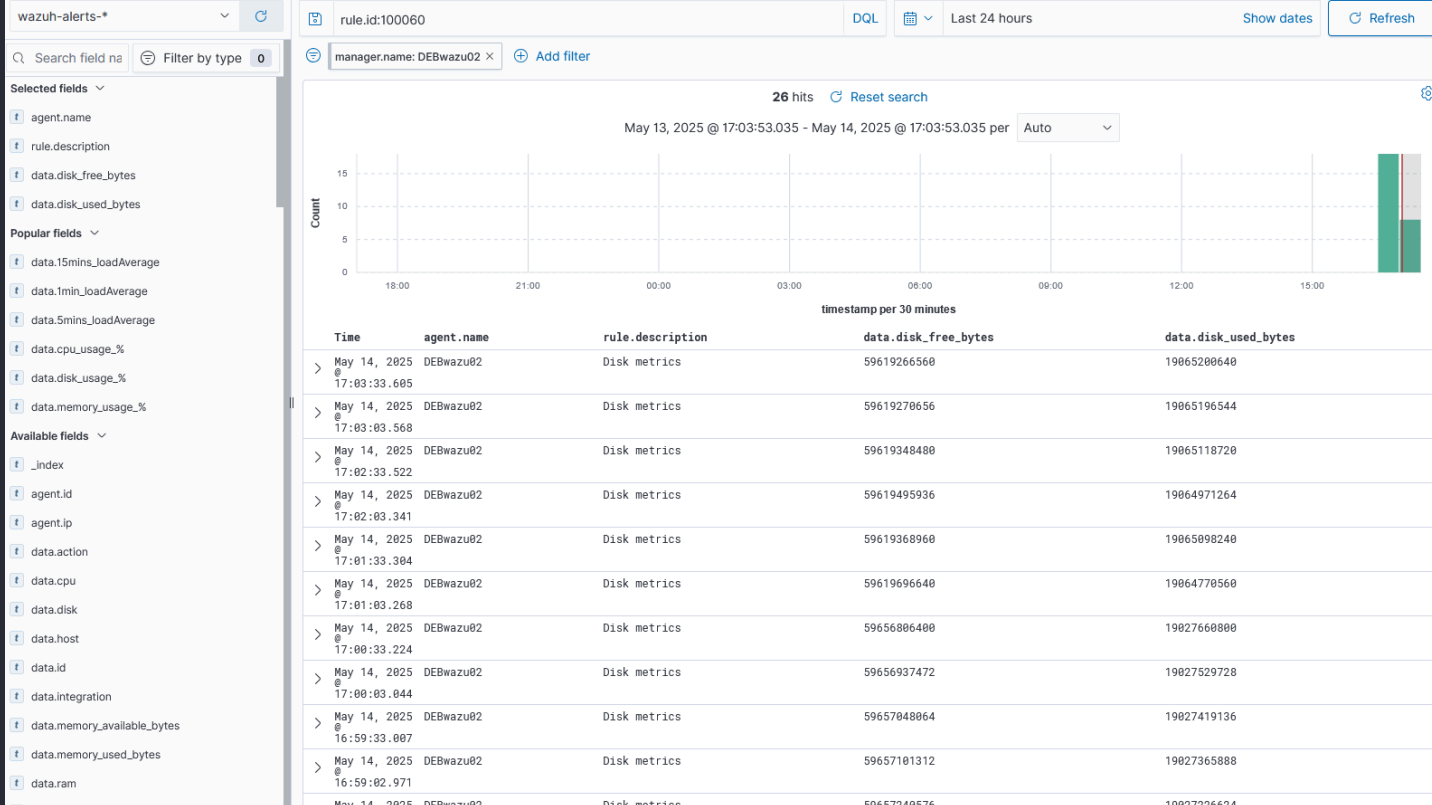
Load Average



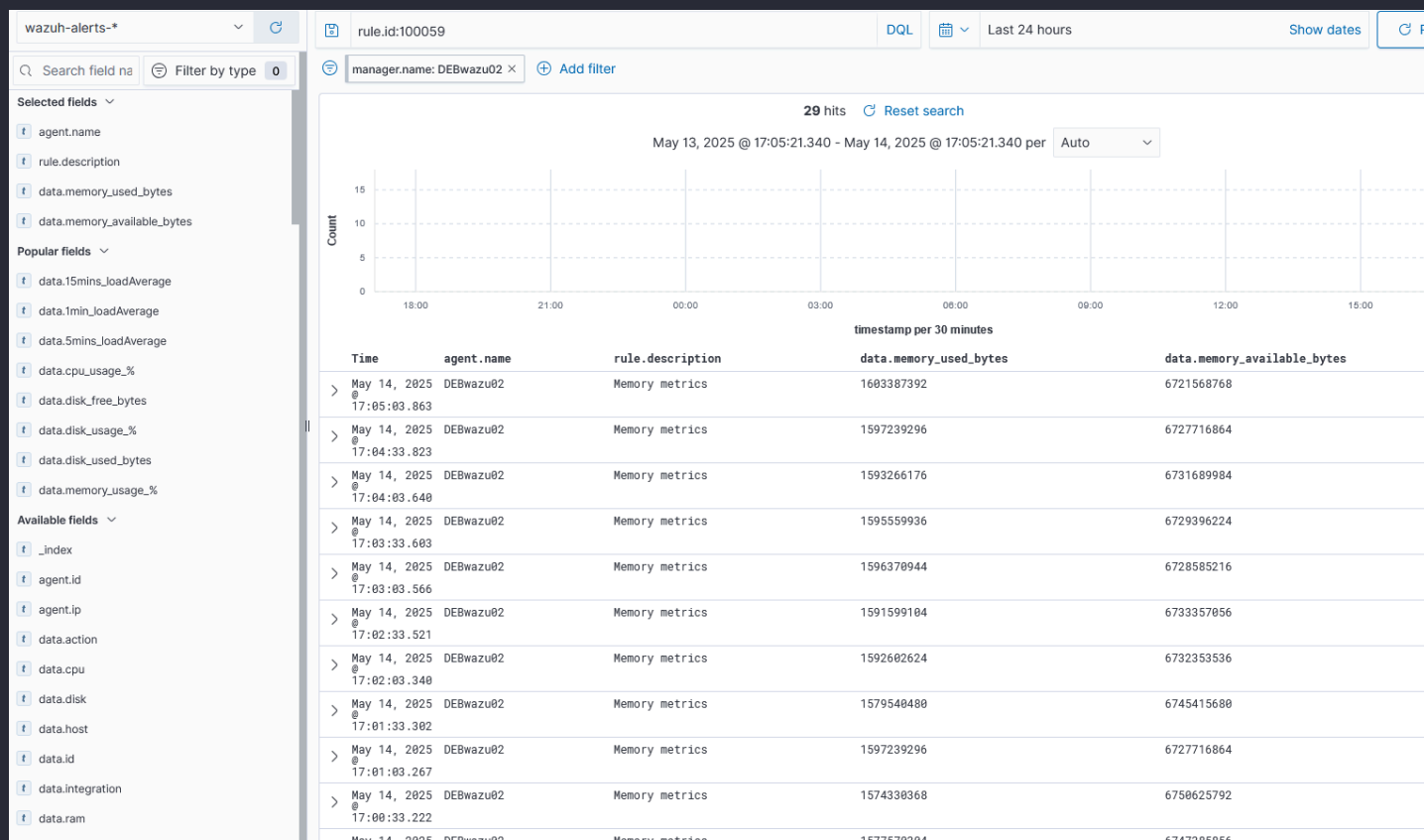
CPU, Memory and Disk metric usage



Disk usage



Memory usage



Custom visualizations and dashboards

Pour utiliser les alertes afin de créer des visualisations et des tableaux de bord, nous devons définir le type de données de tous les champs personnalisés en 'double'. Par défaut, l'indexeur Wazuh analyse les valeurs des alertes existantes comme des types de données 'string' (chaîne de caractères). Pour modifier le type de données par défaut de 'string' à 'double', procédez comme suit :

Wazuh server

1. Ajoutez les champs personnalisés dans le template Wazuh. Trouvez la section "data" dans le fichier `/etc/filebeat/wazuh-template.json`, puis ajoutez les champs personnalisés mis en évidence dans la section "properties" de "data".

```
"1min_loadAverage": {
  "type": "double"
},
"5mins_loadAverage": {
  "type": "double"
},
"15mins_loadAverage": {
  "type": "double"
},
"cpu_usage_%": {
  "type": "double"
},
"memory_usage_%": {
  "type": "double"
},
"memory_available_bytes": {
  "type": "double"
},
"memory_used_bytes": {
  "type": "double"
},
"disk_used_bytes": {
  "type": "double"
},
"disk_free_bytes": {
  "type": "double"
},
"disk_usage_%": {
  "type": "double"
},
```

apply

filebeat setup -index-management

output

```
ILM policy and write alias loading not enabled.
Index setup finished.
```

Il n'est pas permis de modifier les champs existants dans l'indexeur Wazuh. Une fois qu'un index est créé avec certains champs de données, toute modification des champs existants sur l'index en production n'est pas autorisée. Néanmoins, il existe une solution de contournement : le réindexage.

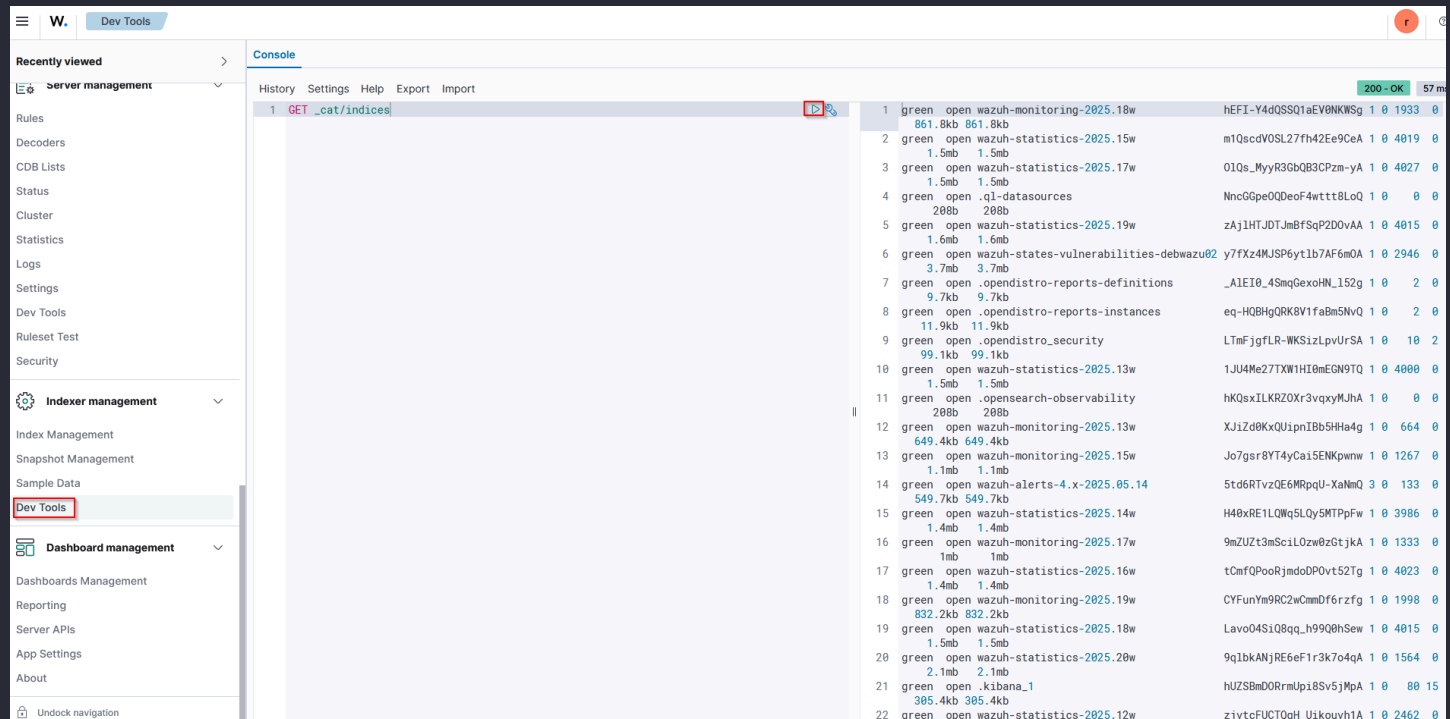
Réindexage

Le réindexage des données est le processus qui consiste à déplacer des données d'un index vers un autre, tout en permettant éventuellement de transformer les données au passage. Cette opération

est réalisée lorsqu'il y a des modifications dans la structure des données ou dans les mappings qui nécessitent de réorganiser ou de mettre à jour l'index.

Wazuh dashboard

1 - Réindexez vos données. Pour utiliser cette technique, nous allons copier les données de l'index d'origine vers un nouvel index avec des définitions de schéma mises à jour. Cliquez sur l'icône du menu en haut à gauche et allez dans Management -> Dev Tools pour accéder à la console. Nous utiliserons cette console pour exécuter les requêtes nécessaires au réindexage des données existantes.

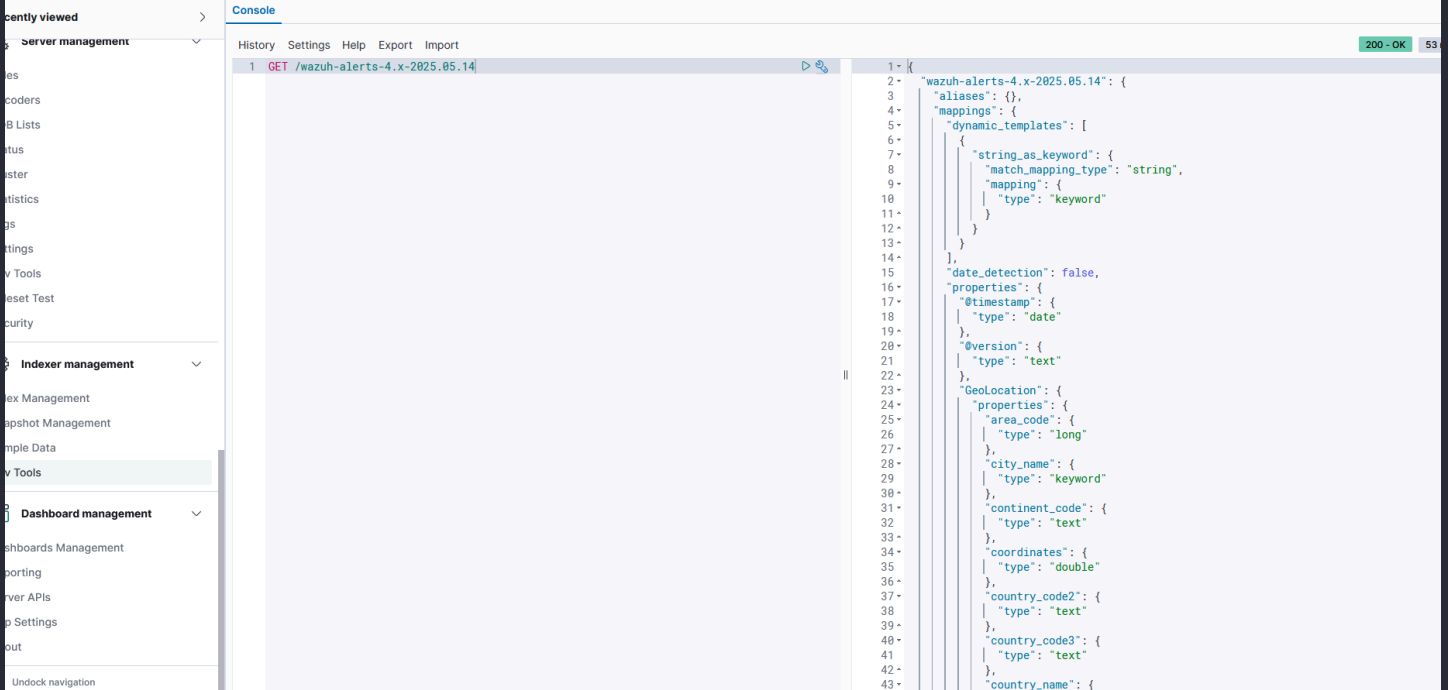


The screenshot shows the Wazuh dashboard interface. On the left, the 'Dev Tools' menu is highlighted under the 'Indexer management' section. The main console area displays a REST client with a GET request to `_cat/indices`. The response is a table of existing indices and their creation dates.

Index Name	Index Type	Index Size	Index Status	Index Creation Date
wazuh-monitoring-2025.18w	green	861.8kb	open	2025.18w
wazuh-statistics-2025.15w	green	1.5mb	open	2025.15w
wazuh-statistics-2025.17w	green	1.5mb	open	2025.17w
.ql-datasources	green	208b	open	2025.19w
wazuh-statistics-2025.19w	green	1.6mb	open	2025.19w
wazuh-states-vulnerabilities-debwazu02	green	3.7mb	open	2025.19w
.opendistro-reports-definitions	green	9.7kb	open	2025.19w
.opendistro-reports-instances	green	11.9kb	open	2025.19w
.opendistro_security	green	99.1kb	open	2025.19w
wazuh-statistics-2025.13w	green	1.5mb	open	2025.13w
.opensearch-observability	green	208b	open	2025.13w
wazuh-monitoring-2025.13w	green	649.4kb	open	2025.13w
wazuh-monitoring-2025.15w	green	1.1mb	open	2025.15w
wazuh-alerts-4.x-2025.05.14	green	549.7kb	open	2025.05.14
wazuh-statistics-2025.14w	green	1.4mb	open	2025.14w
wazuh-monitoring-2025.17w	green	1mb	open	2025.17w
wazuh-statistics-2025.16w	green	1.4mb	open	2025.16w
wazuh-monitoring-2025.19w	green	832.2kb	open	2025.19w
wazuh-statistics-2025.18w	green	1.5mb	open	2025.18w
wazuh-statistics-2025.20w	green	2.1mb	open	2025.20w
.kibana_1	green	305.4kb	open	2025.12w
wazuh-statistics-2025.12w	green	305.4kb	open	2025.12w

shows the names of the existing indices and corresponding creation dates

Récupérez des informations sur l'index depuis l'indexeur Wazuh à l'aide d'une requête GET. Cela permet de confirmer que les champs personnalisés ajoutés sont de type keyword. Dans Wazuh, les index sont créés avec le format `wazuh-alerts-4.x-YYYY.MM.DD`. Ici, nous allons utiliser le dernier index `wazuh-alerts-4.x-2025.05.14`. Assurez-vous de remplacer cette valeur par celle de votre index le plus récent.



Extrait les données de ton index source le plus récent vers le nouvel index de destination nommé wazuh-alerts-4.x-backup en utilisant l'API de réindexation. Remplace le nom de l'index source par la valeur correspondant à ton index le plus récent.

```
POST _reindex
{
  "source": {
    "index": "wazuh-alerts-4.x-2023.04.24"
  },
  "dest": {
    "index": "wazuh-alerts-4.x-backup"
  }
}
```

Suppression des alertes du jour t :

```
DELETE /wazuh-alerts-4.x-2025.05.15
```

Re indexation des alertes grâce au backup :

```
POST _reindex
{
  "source": {
    "index": "wazuh-alerts-4.x-backup"
  },
  "dest": {
    "index": "wazuh-alerts-4.x-2023.04.24"
  }
}
```

Suppression du backup :

```
DELETE /wazuh-alerts-4.x-backup
```

Vérification

```
"memory_available_bytes": {  
  "type": "double"  
},  
"memory_usage_%": {  
  "type": "double"  
},  
"memory_used_bytes": {  
  "type": "double"  
},
```

Dashboard

On peut maintenant manipuler les logs pour en créer des visualisations :

